Statistical Verification of Distributed Programs Within SimGrid

Yann Duplouy

Inria Nancy Grand Est

Monday, May 13th 2019

Context

- SimGrid
- Statistical model-checking
- My previous Work

Current Progress

- BitTorrent example
- Outside scenario generation
- Modifying SimGrid to have a consistent RNG

Conclusion

Distributed Programs

Definition (Distributed Systems)

*A **distributed system** is a collection of components that can interact with one another and may be partly indepent or concurrent.*

Distributed Programs

Definition (Distributed Systems)

*A **distributed system** is a collection of components that can interact with one another and may be partly indepent or concurrent.*

Definition (Distributed Program)

*A **distributed program** is an application that runs on a distributed system.*

What is SimGrid ?

SimGrid is a **software** platform for the **simulation** of distributed programs.

What is SimGrid ?

SimGrid is a **software** platform for the **simulation** of distributed programs.

- Open-source and available for free on `http://simgrid.org`;
- C++ (or Java) **library**;

What is SimGrid ?

SimGrid is a **software** platform for the **simulation** of distributed programs.

- Open-source and available for free on `http://simgrid.org`;
- C++ (or Java) **library**;

Subprogram A

Subprogram B

Subprogram C

What is SimGrid ?

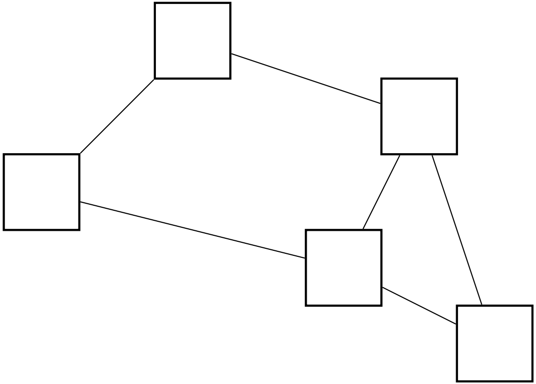
SimGrid is a **software** platform for the **simulation** of distributed programs.

- Open-source and available for free on `http://simgrid.org`;
- C++ (or Java) **library**;

Subprogram A

Subprogram B

Subprogram C



What is SimGrid ?

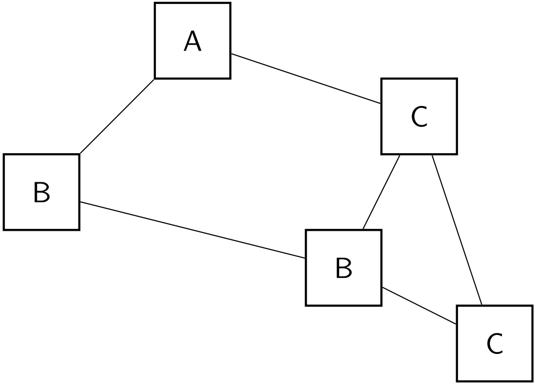
SimGrid is a **software** platform for the **simulation** of distributed programs.

- Open-source and available for free on `http://simgrid.org`;
- C++ (or Java) **library**;

Subprogram A

Subprogram B

Subprogram C



Stochastic models

Problem: «Reality is not perfect»

Idea: Model imperfection by probabilistic laws

Where ?

- Bandwidth of a link;
- Computation power;
- Latency of a link.

Different kinds of analysis

Transient analysis

- What is the probability that at a given moment all computers are busy ?
- How long, in average, does it take for the distributed program to complete ?

Stationary analysis

- What is the *average* energy consumption ?
- What is the probability of a synchronisation error ?

Two methods for model-checking of probabilistic models

Numerical model-checking

- Precise values
- Strong probabilistic hypotheses
- Large memory space

Two methods for model-checking of probabilistic models

Numerical model-checking

- Precise values
- Strong probabilistic hypotheses
- Large memory space

Statistical model-checking

- Confidence interval
- Small memory requirements
- Easy to parallelize
- Weak probabilistic hypothesis
- Requires fully stochastic models
- Rare Event problem

Cosmos

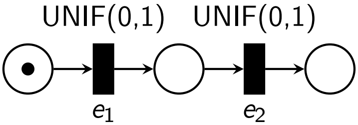
Description [Ballarini, Barbot, Duflot, Haddad, Pekergin 2015]

- Statistical model-checker for HASL over stochastic Petri nets;
- Free software (GPLv3); C++, Ocaml; `http://cosmos.lacl.fr`;
- Developers: Hilal Djafri (2009-2012), Paolo Ballarini (2010-2011), Benoît Barbot (since 2011), Yann duplouy (2015-2018).

Main Applications

- Flexible manufacturing systems;
- Biological networks [Barbot, Kiatkowska 2015];
- Embedded pacemaker model [Barbot, Kwiatkowska, Mereacre, Paoletti 2015].

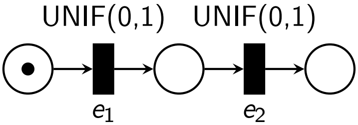
Stochastic PN



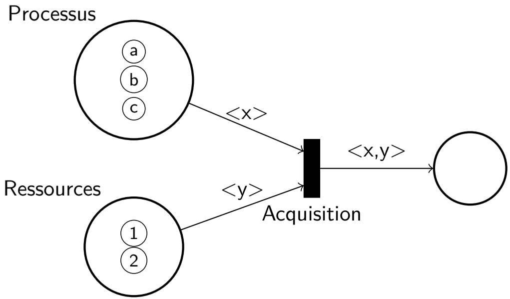
Cosmos

Refresher on Petri nets

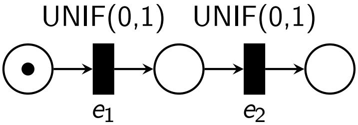
Stochastic PN



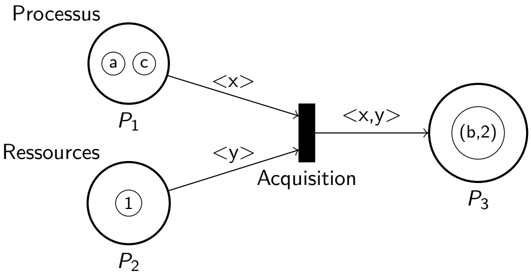
High-level PN



Stochastic PN

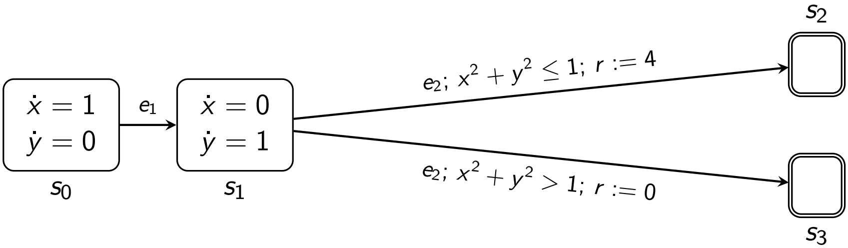


High-level PN



HASL formulas

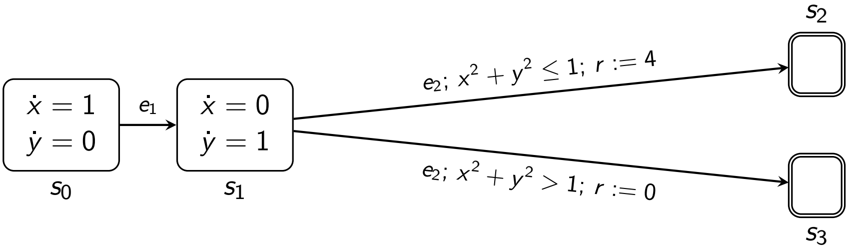
A deterministic hybrid automaton and an expression



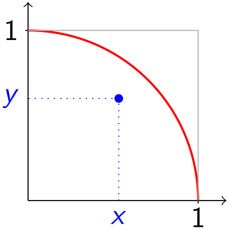
E(**LAST**(**r**))

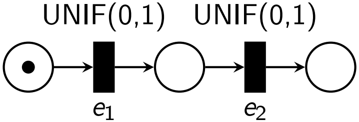
HASL formulas

A deterministic hybrid automaton and an expression



E(**LAST**(**r**))





Context

- SimGrid
- Statistical model-checking
- My previous Work

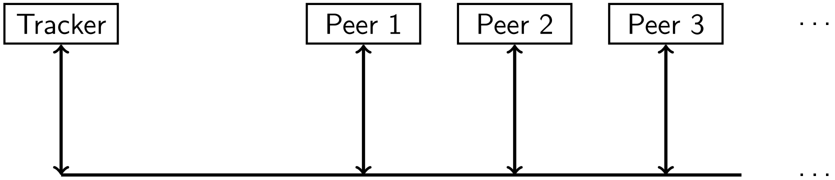
Current Progress

- BitTorrent example
- Outside scenario generation
- Modifying SimGrid to have a consistent RNG

Conclusion

An overview of the BitTorrent protocol

Goal: Deliver a file through a peer-to-peer protocol;



- A **tracker**
- Multiple **peers**, that can be *seeders* or *leechers*

Example available in Simgrid distribution; modified to measure completion time.

Outside scenario generation

Another executable or script:

- Generate environments, given the stochastic description;
- Runs the simulator;
- Gather results from simulations.

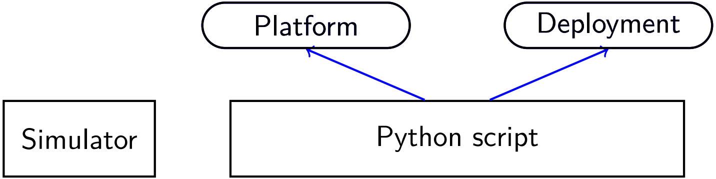
Then we can use R (or other tools) to perform statistical analysis.

Outside scenario generation

Another executable or script:

- Generate environments, given the stochastic description;
- Runs the simulator;
- Gather results from simulations.

Then we can use R (or other tools) to perform statistical analysis.

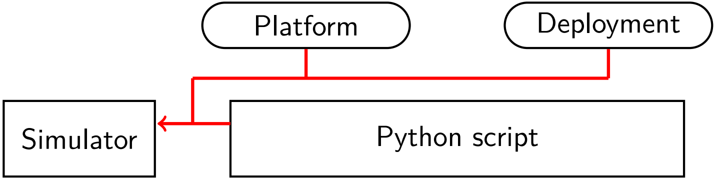


Outside scenario generation

Another executable or script:

- Generate environments, given the stochastic description;
- Runs the simulator;
- Gather results from simulations.

Then we can use R (or other tools) to perform statistical analysis.

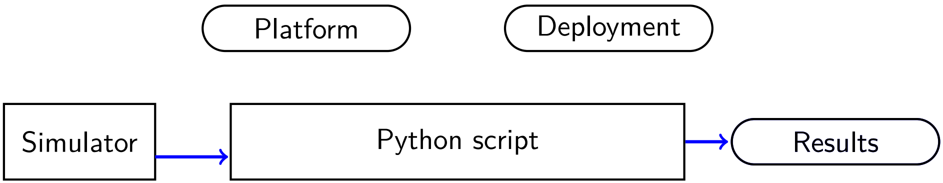


Outside scenario generation

Another executable or script:

- Generate environments, given the stochastic description;
- Runs the simulator;
- Gather results from simulations.

Then we can use R (or other tools) to perform statistical analysis.

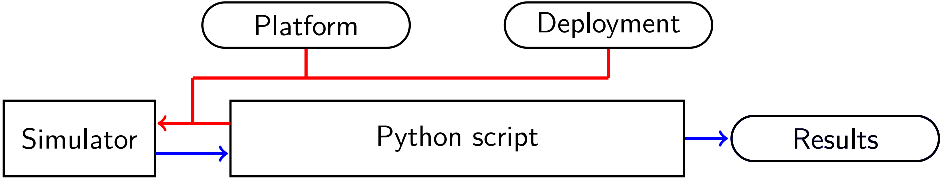


Outside scenario generation

Another executable or script:

- Generate environments, given the stochastic description;
- Runs the simulator;
- Gather results from simulations.

Then we can use R (or other tools) to perform statistical analysis.



Quick results 1/2

Measuring *average* completion time (with 95% confidence level)

		leechers	
		25MB/s	$U_{\texttt{int}}(1024, 8191)$kB/s
seeders	25 MB/s	324s	704s
	$U_{\texttt{int}}(1024, 8191)$ kB/s	321s	710s

Quick results 2/2

Introducing failures, generated up to 600 000 seconds:

1. Host becomes unavailable after EXP(1000)
2. Host becomes availables again after UNIF(10, 20), repeat (1.)

All peers are connected to the backbone at 4 kBps.

Quick results 2/2

Introducing failures, generated up to 600 000 seconds:

1. Host becomes unavailable after EXP(1000)
2. Host becomes availables again after UNIF(10, 20), repeat (1.)

All peers are connected to the backbone at 4 kBps.

- if only seeders have failures:
  most (99%) simulations run under 660s,
  average simulation time 613s;
  but slowest simulation took 12 209 seconds;
- if seeders and leechers have failures:
  most (99%) simulations run under 704s,
  average simulation time 850s;
  but slowest took 623 624 seconds;

A cleaner approach ?

Add a few SimGrid modules:

- **Modify** the *profile* class to accept stochastic definitions
- **Implement** a statistical verification class:
  Measure an approximation of performance indexes
  Allow to restart simulations without multiple external calls to the simulator

Specification of probabilities

- **Profiles** describe how capacities *deterministically* change;

Specification of probabilities

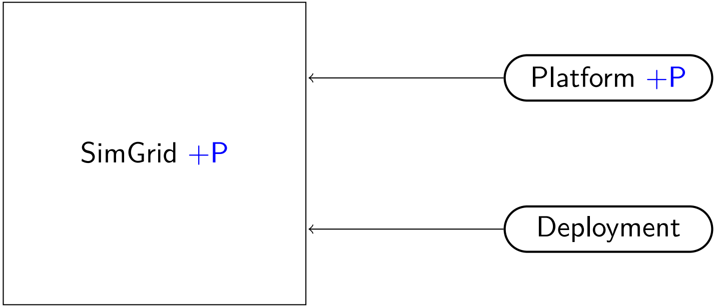
- **Profiles** describe how capacities *deterministically* change;
- $\Rightarrow$ **Stochastic Profiles** allow for *stochastic* descriptions.

Specification of probabilities

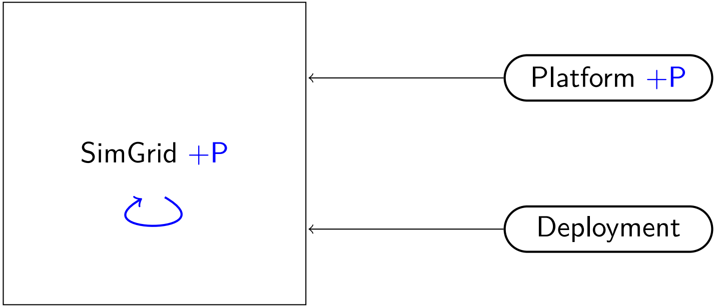
- **Profiles** describe how capacities *deterministically* change;
- $\Rightarrow$ **Stochastic Profiles** allow for *stochastic* descriptions.

Platforms now accept stochastic profiles.

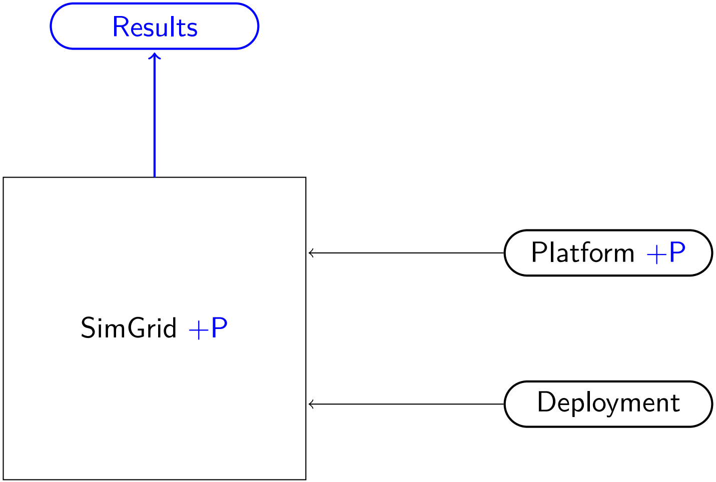
Running multiple times



Running multiple times



Running multiple times



Context
SimGrid
Statistical model-checking
My previous Work

Current Progress
BitTorrent example
Outside scenario generation
Modifying SimGrid to have a consistent RNG

Conclusion

Conclusion

SimGrid is good for *reproducible* scenarios, but real scenarios are unpredictable

- Generating scenarios and deterministic profiles is inconvenient at best;
- Adding a module dedicated to statistical verification is a cleaner approach;
- Modifying the *profile* class is a first step;
- Restarting simulation properly is currently in progress;
- Implementing HASL into SimGrid would increase hugely the expressivity.